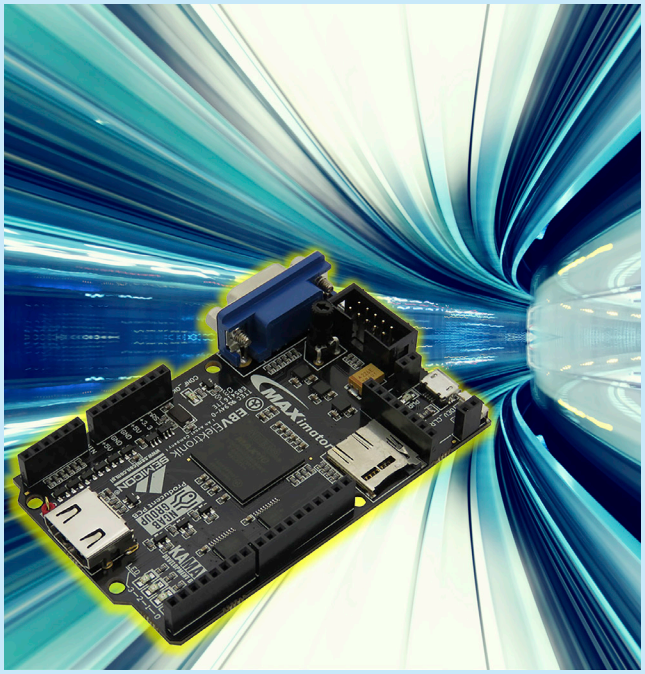
Podstawy generowania grafik w FPGA za pomocą VHDL (2)

Implementacja projektu



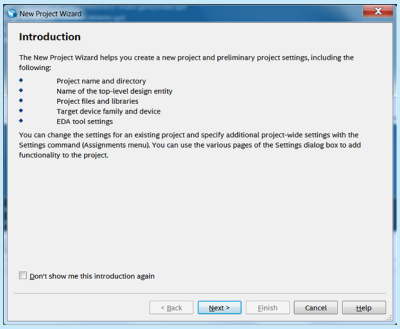
W tym odcinku przechodzimy do tworzenia krok po kroku projektu, który w ostatecznym rozrachunku stanie się generatorem prostych elementów graficznych. Do implementacji sprzętowych użyjemy zestawu maXimator, który wyposażono między innymi w złącze VGA.

W pierwszej kolejności należy upewnić się, że dysponujemy najnowszą wersją oprogramowania Quartus Prime Lite. Wersję instalacyjną programu można pobrać spod adresu ***https://goo.gl/5seu1Q***. Należy wybrać program „Quartus Prime software Lite edition" (dla potrzeb tego artykułu używano wersji 16.1). Przed pobraniem należy założyć konto, które jest darmowe. Oprogramowanie po rozpakowaniu zajmuje nieco ponad 13 GB, dlatego wcześniej należy upewnić się, że dysponujemy wolnym miejscem na dysku komputera. Po poprawnym pobraniu i zainstalowaniu uruchamiamy program – na ekranie powinno zostać wyświetlone okno jak **na rysunku 19.** Po jego otwarciu z menu *File* wybieramy *New Project Wizard* – zostanie wyświetlone okno kreatora pokazane na **rysunku 20**.

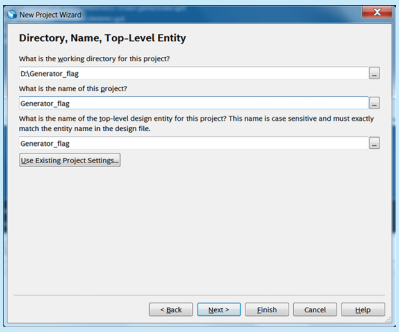
Klikamy w przycisk *Next*. Zostanie wyświetlone okno, w którym należy podać nazwę projektu oraz jego lokalizację na dysku. Nasz przykładowy projekt będzie się nazywał *Generator_flag* i zostanie zapamiętany w folderze *Generator_flag* na dysku D (**rysunek 21**).

Ponownie klikamy *Next*. Jeśli zostanie wyświetlony komunikat pokazany na **rysunku 22**, naciskamy *Yes*. Komunikat informuje nas tylko o tym, że na dysku jest tworzony nowy folder. W dalszej kolejności powinno zostać wyświetlone okno, jak na **rysunku 23**.

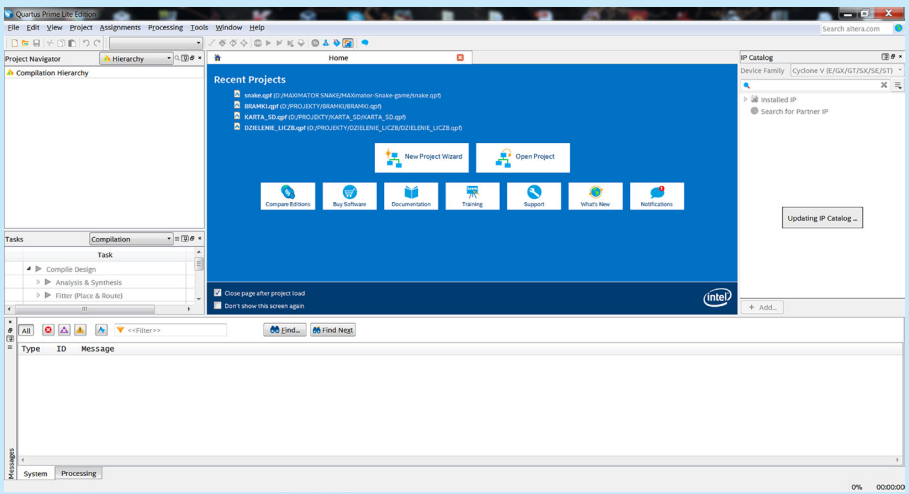
W oknie tym niczego nie zmieniamy – należy pozostawić zaznaczoną opcję *Empty project* i kliknąć *Next*. Następnie zostanie wyświetlone okno jak na **rysunku 24**. Tu też niczego nie zmieniamy – klikamy *Next*. Po jego



Rysunek 20. Okno kreatora projektu – rozpoczęcie tworzenia nowego projektu



Rysunek 21. Okno kreatora – wprowadzenie nazw folderu, projektu i pliku głównego



Rysunek 19. Ekran główny środowiska Quartus Prime

wybraniu zostanie pokazane najistotniejsze z okien kreatora, w którym będziemy wybierali typ układu, w który jest wyposażony zestaw maXimator (**rysunek 25**).

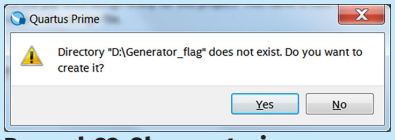
Jeśli zostanie wyświetlone pomniejszone okno, dla wygody warto je powiększyć, chwytając myszką za dolną krawędź okna i przesuwając ją. W oknie należy rozwinąć listę *Family* i wybrać „MAX 10 (DA/DE/DC/SA/SC)". W wyniku tego wyboru na dole okna zostanie wyświetlona lista układów, z której wybieramy „10M08DAF256C8GES". Należy zaznaczyć układ, jak na **rysunku 26** i kliknąć *Next*.

Jako kolejne zostanie wyświetlone okno umożliwiające wybranie narzędzi symulacyjnych oraz oprogramowania, którego środowisko może użyć do przenoszenia programu na układ (**rysunek 27**). My nie będziemy go potrzebowali w tym projekcie, więc nie wykonując żadnych zmian, po prostu klikamy *Next*. W wyniku kliknięcia na ekranie zostanie wyświetlone podsumowania kreatora projektu, które powinno wyglądać jak na **rysunku 28**.

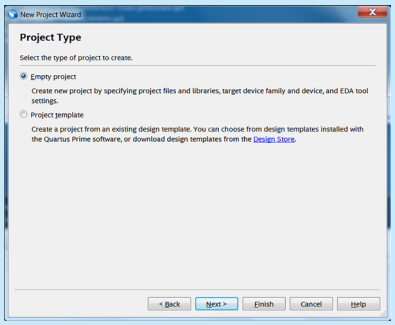
Jeśli wyświetlone informacje są poprawne, to klikamy *Finish*. W wyniku tego rozpocznie się tworzenie projektu. Po zakończeniu pracy kreatora na ekranie komputera zostanie wyświetlony obraz pokazany na **rysunku 29**.

Dla wygody, aby zwiększyć ilość miejsca dostępnego na ekranie, możemy zamknąć okno znajdujące się po prawej związane z *IP Catalogiem*.

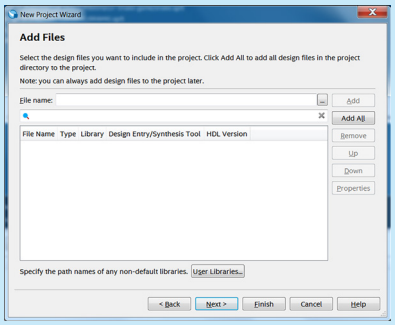
W wyniku pracy kreatora utworzyliśmy projekt, który nie zawiera żadnych plików (zgodnie z opcją *Empty project*, którą wybraliśmy w kreatorze projektu a która dosłownie powoduje utworzenie projektu bez jakichkolwiek plików bibliotecznych) – w razie potrzeby trzeba je dodać ręcznie. Pierwszym, jaki dodamy, będzie



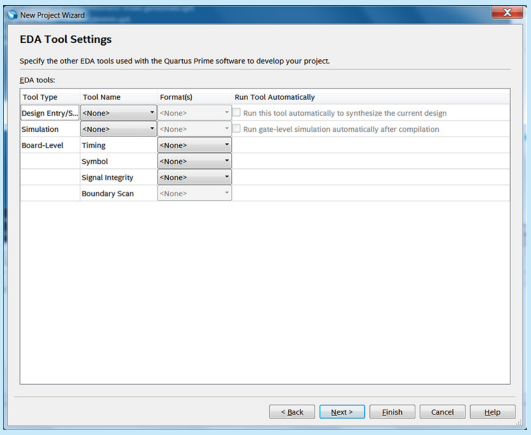
Rysunek 22. Okno z pytaniem o utworzenie nowego pliku



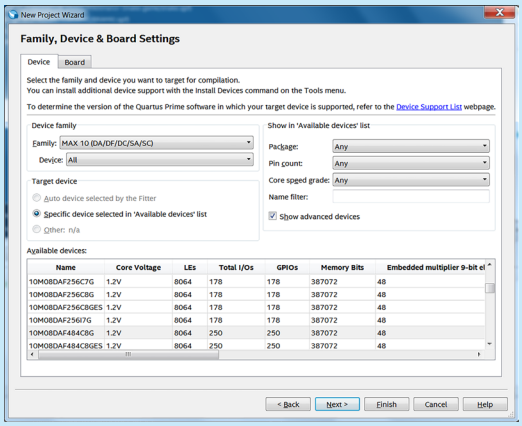
Rysunek 23. Okno kreatora – wybór typu projektu



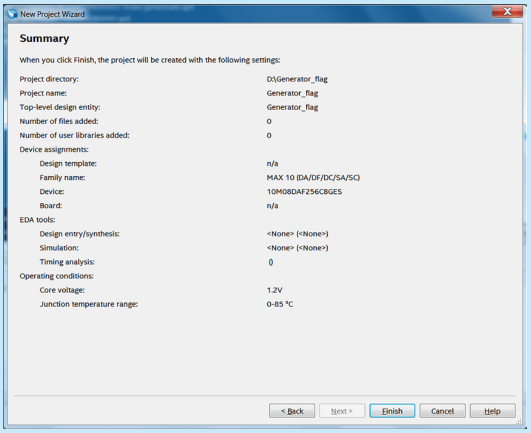
Rysunek 24. Okno kreatora – wybór plików dodawanych do projektu



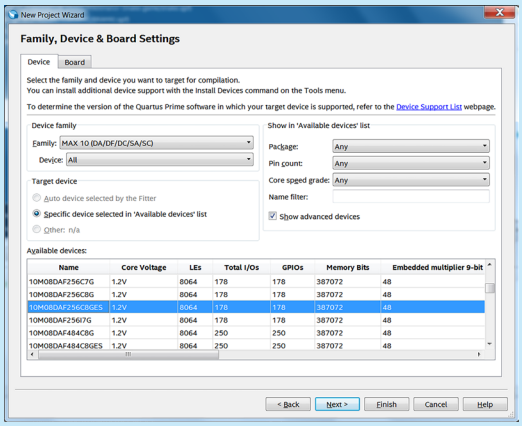
Rysunek 27. Okno kreatora – wybór narzędzi projektowych



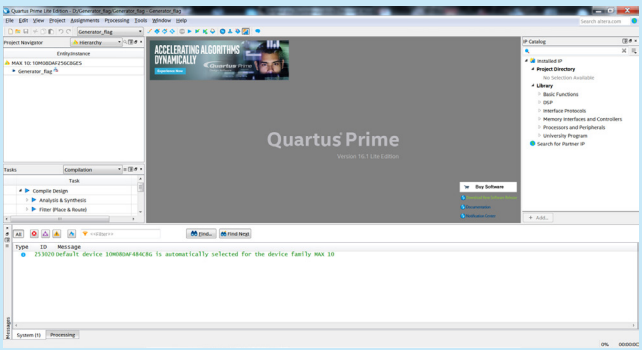
Rysunek 25. Okno kreatora – wybór rodziny układów



Rysunek 28. Okno kreatora – podsumowanie wybranych opcji



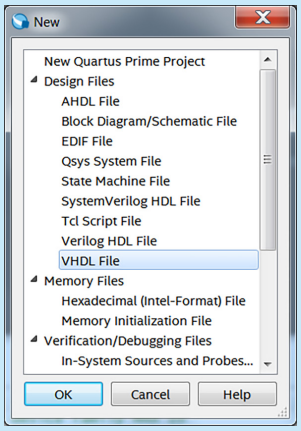
Rysunek 26. Okno kreatora – zaznaczenie wyboru układu FPGA



Rysunek 29. Ekran środowiska Quartus Prime po zakończeniu pracy kreatora projektu

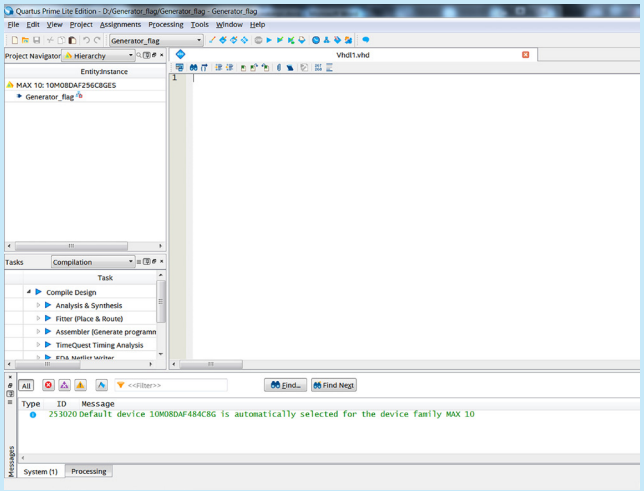
plik, w którym zawrzemy zasadniczą część naszego generatora flag. W tym celu należy kliknąć w ikonę z białą kartką lub wybrać *File* → *New*, względnie użyć skrótu klawiszowego CTRL+N. W rezultacie zostanie wyświetlone okno jak na **rysunku 30**. W oknie tym zaznaczamy *VHDL File*, a następnie klikamy *OK*. W ten sposób tworzymy pusty plik, do którego wpiszemy kod programu (**rysunek 31**).

Pliku nie można zapamiętać na dysku dopóty, dopóki jest pusty. W celu jego zapamiętania należy wprowadzić dowolny znak – może to być na przykład znak odstępu (spacja). Dla porządku naciśnijmy spację i od razu zapamiętajmy „edytowany" tak plik. W tym celu klikamy na ikonę z dyskietką lub z menu wybieramy *File* → *Save As* i zapisujemy plik w folderze z projektem (w tym przykładzie jest to *D:\Generator_flag*) za pomocą okna dialogowego pokazanego na **rysunku 32**. Naszemu plikowi z przykładu nadajmy taką samą nazwę, jaką ma projekt *Generator_flag*, dodając rozszerzenie nazwy *.vhd*. Uwaga – nazwa nie może być dowolna, gdyż w projekcie musi istnieć plik główny, który musi mieć identyczną nazwę jak projekt. W przeciwnym razie Quartus Prime Lite będzie miał problem z obsługą projektu. Po wykonaniu wymienionych czynności klikamy *Zapisz*. Tym samym mamy już przygotowany projekt z plikiem o nazwie takiej, pod jaką go przed chwilą zapamiętaliśmy.

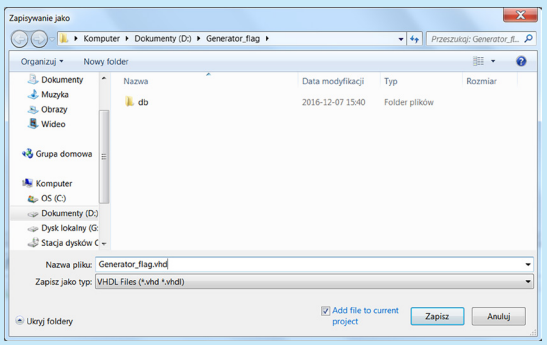


Rysunek 30. Wybór typu edytowanego pliku

Teraz możemy przystąpić do pisania właściwego kodu związanego z generatorem flag.



Rysunek 31. Okno główne środowiska po utworzeniu pustego pliku VHDL



Rysunek 32. Wpisanie nazwy tworzonego pliku źródłowego

W pierwszej kolejności musimy dodać biblioteki, które będą używane w projekcie. W tym celu za pomocą edytora tekstowego w pliku głównym wpisujemy następujące:

```
library IEEE;
use ieee.std_logic_1164.all;
```

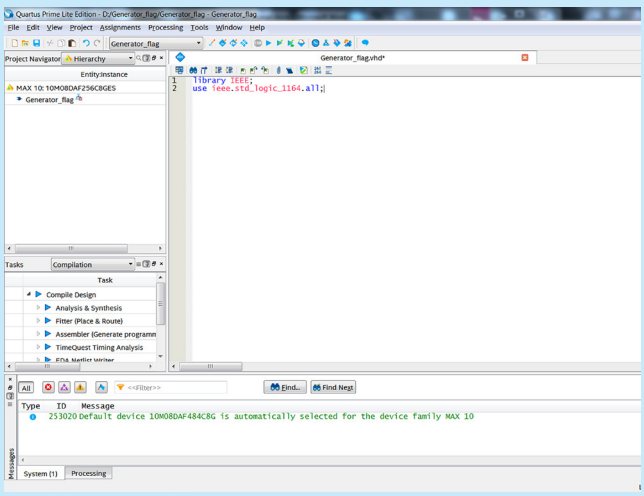
W ten sposób importujemy do projektu bibliotekę IEEE, która zawiera definicje typów zmiennych, sygnałów, podstawowe funkcje konwersji itp. Jednym zdaniem – importujemy wszystko, co jest potrzebne do tworzenia programów w języku VHDL. Trzeba przy tym zauważyć, że tej biblioteki nie importujemy w całości – wybieramy tylko jej poszczególne części składowe, czyli paczki (*packages*). Do naszego projektu importujemy jedną paczkę – *std_logic_1164* – zawierającą definicje wszelkich typów i podtypów używanych w języku VHDL. Jest to biblioteka, która będzie przypuszczalnie przez nas używana w większości projektów. Kompilator utożsamia małe i duże litery, więc nie trzeba się przejmować wielkością znaków. Z drugiej strony warto wypracować sobie jakiś jednolity styl pisania programu, dopasować się do istniejących standardów, aby nie mieć problemu chociażby z analizowaniem własnych czy cudzych programów. Po wprowadzeniu zmian przykładowy plik z projektu powinien wyglądać jak na **rysunku 33**.

Po dołączeniu do projektu bibliotek należy przystąpić do skonfigurowania wejść/wyjść układu FPGA. W tym projekcie przyda się nam znajomość standardu VGA – złącze monitora VGA z opisem dostępnych sygnałów pokazano na **rysunku 34**. Choć standardowe złącze VGA zawiera 15 wyprowadzeń, to w praktyce do poprawnego działania wystarczy 9. Na przytoczonym rysunku są to wyprowadzenia:

- 1 – sygnał koloru czerwonego,
- 2 – sygnał koloru zielonego,
- 3 – sygnał koloru niebieskiego,
- 13 – impulsy synchronizacji poziomej,
- 14 – impulsy synchronizacji pionowej,
- 5,6,7 i 8 – masa ogólna i masy dla poszczególnych sygnałów kolorów.

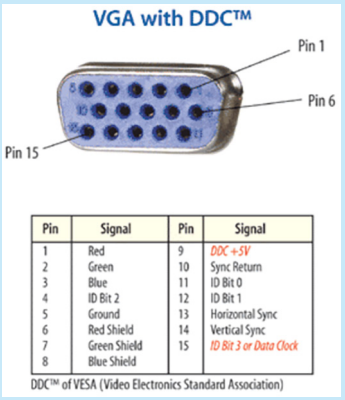
Dzięki sygnałom złącza VGA możemy zdefiniować potrzebne wyjścia. Oprócz nich będzie potrzebne nam jedno wejście, związane z „zegarem graficznym", który będzie taktował generator flag. W jakim celu jest nam potrzebny?

Gdy rozpatrywaliśmy matrycę złożoną z diod LED, napisaliśmy, że aby otrzymać wrażenie świecenia całej matrycy, należy bardzo szybko gasić i zaświecać kolejne LEDy w wierszach takiej matrycy bądź też przez ten sam czas poszczególne diody pozostawić wyłączone. Jednak trzeba robić to bardzo szybko, w ułamkach sekundy, aby ludzkie oko nie zauważyło migotania. Tym włączaniem i wyłączaniem steruje wspomniany przebieg wejściowy o częstotliwości,



Rysunek 33. Wpisanie poleceń dołączających biblioteki IEEE

która jest ściśle związana z rozdzielczością pokazywanego obrazu oraz częstotliwością jego odświeżania. Inaczej mówiąc – czas trwania pojedynczego impulsu jest równy czasowi przeznaczonemu na świecenie lub zgaszenie diody. Chociaż raczej będziemy mieli do czynienia po prostu z pikselami zbudowanymi z innych elementów, to jednak większość standardów wyświetlania obrazu (w tym VGA) bazuje na takim sposobie jego tworzenia.



Rysunek 34. Złącze VGA (źródło: https://goo.gl/JYol9y)

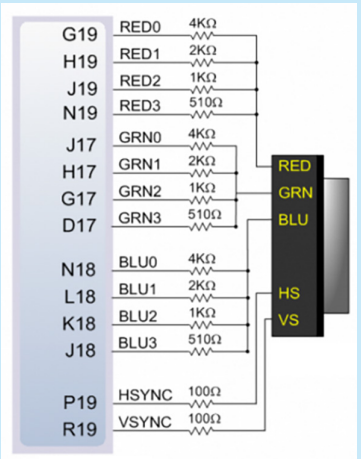
Skoro już wiemy, jakich wejść i wyjść potrzebujemy, to zdefiniujmy je w kodzie programu. Nie musimy się przy tym przejmować połączeniami mas sygnałowych, ponieważ odpowiednie połączenia wykonano na płytce drukowanej. Generując sygnały zgodne z wymaganiami standardu VGA, można użyć przetwornika, który jest złożony z zestawu rezystorów połączonych z pojedynczymi, wspólnymi punktami, a którego poglądowy schemat pokazano na **rysunku 35**. Za pomocą zestawu rezystancji są kodowane sygnały barwne R, G, B, natomiast impulsy synchronizacji poziomej i pionowej są przesyłane jedynie z użyciem pojedynczego rezystora. W standardzie VGA jest konieczna zamiana kombinacji binarnych na poziomy napięcia zgodnie z zależnością, że im większa liczba, tym wyższe napięcie dla danej barwy. Poziom napięcia odpowiada jasności piksela na ekranie – im wyższe, tym piksel jest jaśniejszy. Jak łatwo się domyślić, napięcie poszczególnych składowych koloru odpowiada za wypadkowy kolor piksela wyświetlanego na ekranie. Inaczej jest w wypadku impulsów synchronizacji – tu jest jedynie wymagane napięcie zapewniające umowną „jedynkę" i „zero" dla układu generatora impulsów synchronizacji.

W zestawie maXimator nie ma takiego przetwornika jak opisywany. Jego twórcy zdecydowali się jedynie na zastosowanie ograniczających prąd, dopasowujących rezystorów szeregowych. Tym samym na jedno wyprowadzenie sygnału barwy będzie przypadało jedno wyjście. Trzeba też przy tym zauważyć (napisano również o tym w dokumentacji płytki maXimator), że bez stosowania dodatkowych układów pozwala to na wyświetlenie 8 kolorów na ekranie monitora.

Podsumowując spostrzeżenia, możemy stwierdzić, że budowany przez nas generator flag wymaga użycia:

- 5 wyjść (jedno wyprowadzenie na każdą barwę oraz po jednym wyprowadzeniu dla impulsów synchronizacji pionowej i poziomej),
- 1 wejście dla przebiegu taktującego „zegar graficzny".

Dla potrzeb generowania sygnału synchronizacji poziomej jest potrzebne wyjście, jednak z tej racji, że będziemy je wykorzystywać w kodzie (wyjaśnimy to w kolejnej części artykułu), musimy je skonfigurować jako port wejścia/wyjścia. Pozostałe sygnały należy zadeklarować jako wyjścia.



Rysunek 35. Przykładowy przetwornik rezystorowy (źródło: https://goo.gl/fmdtr3)

Mając już wszystko, co potrzebne, dla czytelności programu pozostawiamy jedną pustą linijkę po wprowadzonym wcześniej tekście i wprowadzamy następujące deklaracje:

```
ENTITY Generator_flag is
    PORT (Zegar: IN STD_LOGIC;
          VGA_HS: INOUT STD_LOGIC;
          VGA_VS,VGA_R,VGA_G,VGA_B
:OUT STD_LOGIC));
End VGA;
```

Pokazany fragment kodu programu odzwierciedla dokładnie to, co omówiliśmy wyżej. Widzimy, że w celu zdefiniowania:

- Wejścia używamy słowa „IN".
- Wejścia/wyjścia słowa „INOUT".
- Wyjścia używamy słowa „OUT".

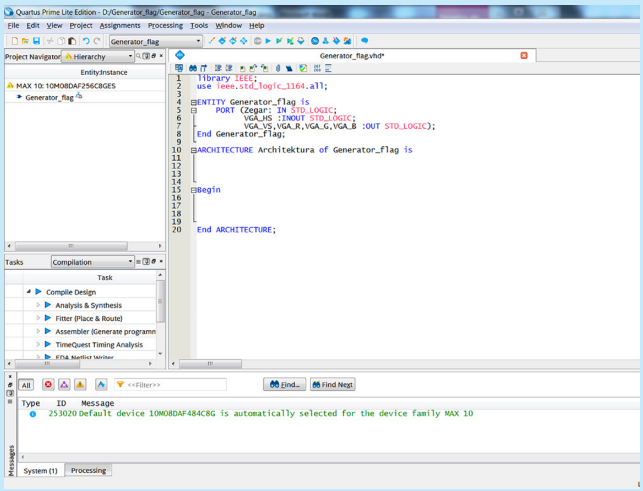
Ponadto dla pojedynczego wejścia/wyjścia używamy typu „STD_LOGIC", natomiast dla wejść/wyjść grupujących kilka odrębnych używamy typu „STD_LOGIC_VECTOR", podając w nawiasie rozmiar takiego wejścia i kolejność występowania w nim bitów. Na przykład rozmiar wyjść opisanych „2 downto 0" wynosi 3, ponieważ są to wyjścia 3-bitowe, w których kolejność bitów jest rozpatrywana od najstarszego do najmłodszego, o czym mówi nam słowo „downto".

Mamy już zdefiniowane wszystkie potrzebne wejścia i wyjścia, co jednak z zachowaniem się układu? Musimy jeszcze określić to, jak ma „pracować". W tym celu należy wprowadzić następujący tekst:

```
ARCHITECTURE Architektura of Generator_flag is
        .
        .
        .
    Begin
        .
        .
        .
End ARCHITECTURE;
```

Jest blok, w którym będzie zawarty kod programu realizującego nasz generator flag. Wobec tego, pozostawiając dla czytelności wolną linijkę po poprzednim kodzie, wprowadź. Wygląd ekranu środowiska Quartus po wpisaniu wspomnianego tekstu pokazano na **rysunku 36**.

Jakub Tyburski, WAT



Rysunek 36. Wygląd ekranu Quartus Prime po wprowadzeniu tekstu programu z artykułu

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 44747, pass: 3qwdwa8u